

AD-A253 324



Technical Report

CMU/SEI-92-TR-2

ESD-92-TR-2

2



Carnegie-Mellon University

Software Engineering Institute

**The Conceptual Basis  
for a Project Support Environment  
Services Reference Model**

Alan W. Brown

Peter H. Feller

January 1992

DTIC  
SELECTE  
JUL 30 1992  
S B D

DISTRIBUTION STATEMENT A

Approved for public release  
Distribution Unlimited

92-20299



9292 7 27

215  
215

The following statement of assurance is more than a statement required to comply with the federal law. The law requires that all people are included in the federal financial aid program. Carnegie Mellon University assures that all people are included in the federal financial aid program. Carnegie Mellon University assures that all people are included in the federal financial aid program. Carnegie Mellon University assures that all people are included in the federal financial aid program.

Carnegie Mellon University does not discriminate and Carnegie Mellon University is required not to discriminate against any person on the basis of race, color, national origin, sex or handicap in violation of Title VI of the Civil Rights Act of 1964, Title IX of the Educational Amendments of 1972 and section 504 of the Rehabilitation Act of 1973 or other federal, state or local laws or executive orders. In addition, Carnegie Mellon does not discriminate in admissions and employment on the basis of religion, creed, ancestry, belief, age, veteran status or sexual orientation in violation of any federal, state or local laws or executive orders. Any person in application of this policy should be directed to the Provost, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, telephone 412/268-4444 or the Vice President for Enrollment, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, telephone 412/268-4444.

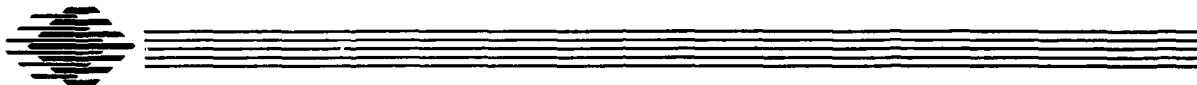
**Technical Report**

**CMU/SEI-92-TR-2**

**ESD-92-TR-2**

**January 1992**

**The Conceptual Basis  
for a Project Support Environment  
Services Reference Model**



**Alan W. Brown**

**Peter H. Feiler**

Software Development Environments Project

Approved for public release.  
Distribution unlimited.

**Software Engineering Institute**  
Carnegie Mellon University  
Pittsburgh, Pennsylvania 15213

This technical report was prepared for the

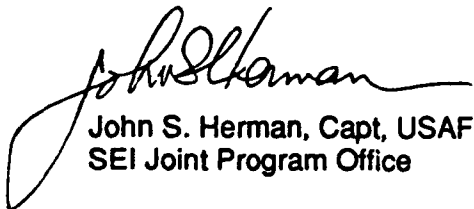
SEI Joint Program Office  
ESD/AVS  
Hanscom AFB, MA 01731

The ideas and findings in this report should not be construed as an official DoD position. It is published in the interest of scientific and technical information exchange.

#### **Review and Approval**

This report has been reviewed and is approved for publication.

FOR THE COMMANDER



John S. Herman, Capt, USAF  
SEI Joint Program Office

The Software Engineering Institute is sponsored by the U.S. Department of Defense.

This report was funded by the U.S. Department of Defense.

Copyright © 1992 by Carnegie Mellon University.

This document is available through the Defense Technical Information Center. DTIC provides access to and transfer of scientific and technical information for DoD personnel, DoD contractors and potential contractors, and other U.S. Government agency personnel and their contractors. To obtain a copy, please contact DTIC directly: Defense Technical Information Center, Attn: FDRA, Cameron Station, Alexandria, VA 22304-6145.

Copies of this document are also available through the National Technical Information Service. For information on ordering, please contact NTIS directly: National Technical Information Service, U.S. Department of Commerce, Springfield, VA 22161.

Copies of this document are also available from Research Access, Inc., 3400 Forbes Avenue, Suite 302, Pittsburgh, PA 15213.

Use of any trademarks in this report is not intended in any way to infringe on the rights of the trademark holder.

# Table of Contents

<b>1. Introduction</b>	<b>1</b>
1.1. Motivation	2
1.2. Our View of a PSE	2
<b>2. Related Work</b>	<b>5</b>
2.1. Existing PSEs, Standards, and Reference Models	5
2.2. Toward an Understanding of Integration	6
<b>3. Approach Taken</b>	<b>7</b>
<b>4. Ways of Using This Reference Model</b>	<b>9</b>
<b>5. The Basic Concepts of the Model</b>	<b>11</b>
5.1. The PSE Services Reference Model	11
5.1.1. PSE Infrastructure Services	12
5.1.2. PSE End-User Services	13
<b>6. Toward an Understanding of Service Interfaces</b>	<b>17</b>
6.1. PSE End-User Service Interfaces	17
6.2. PSE Infrastructure Service Interfaces	18
6.3. Tools and Environment Architectures	19
6.4. Process Issues	19
<b>7. Summary and Conclusions</b>	<b>21</b>
<b>Appendix A Detailed Breakdown of PSE Services</b>	<b>23</b>
<b>Glossary</b>	<b>27</b>
<b>References</b>	<b>29</b>

<b>Accession For</b>	
NTIS CRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	



## List of Figures

<b>Figure 5-1</b>	<b>The Main Components of the PSE Services Reference Model</b>	<b>11</b>
<b>Figure 5-2</b>	<b>The PSE Infrastructure Services</b>	<b>12</b>
<b>Figure 5-3</b>	<b>The PSE End-User Services</b>	<b>14</b>
<b>Figure 6-1</b>	<b>Adding Process Aspects to the PSE Services Reference Model</b>	<b>20</b>





# **The Conceptual Basis for a Project Support Environment Services Reference Model**

**Abstract:** The foundation for a Project Support Environment (PSE) services reference model is presented. This model is to be used as the basis for understanding more about the meaning of integration in a PSE, comparing and contrasting PSE tools and products, and for helping in the identification of PSE interface areas that are candidates for standardization.

The model views a PSE as a set of services, distinguishing between services as perceived by PSE end-users, and those provided as mechanisms within the PSE infrastructure. Process constraints on those services are separately identified. The motivation for this view of a PSE is described, followed by a detailed description of the main structure and elements of the model.

## **1. Introduction**

This paper describes the conceptual basis for a Project Support Environment (PSE) services reference model that we are developing at the SEI. The aim in developing this model is to provide a conceptual framework that helps in understanding a number of issues with regard to PSEs—their role and functionality, the relationships between existing and proposed PSE products, and the position of various PSE standards efforts within an overall PSE context.

One particular use of the concepts embodied in this PSE services reference model is as an aid to identifying PSE interface areas where standardization may improve our ability to build a PSE from existing off-the-shelf products. This is the primary aim of the Project Support Environment Standards Working Group (PSESWG) of the U.S. Navy's Next Generation Computing Resources (NGCR) initiative. The concepts and the PSE services reference model described in this report are being used as the basis for a PSESWG PSE reference model that will help them in identifying relevant interface areas.<sup>1</sup>

This report provides the conceptual basis for a PSE services reference model, but does not provide a complete description of a reference model that would result from applying these concepts. A complete description of the reference model would consist of an explanation of the concepts, a detailed description of each of the services that make up the reference model, and a discussion of the relationship between the concepts and the service descriptions. In this report we concentrate on the first of these three elements, together with a brief description of an example set of services and their relationship to the defining concepts. A current task of the PSESWG is to produce a complete and detailed reference model description by adapting the concepts described here, and then generating a substantial set of service descriptions [9]. We are actively participating in the PSESWG to help them in achieving this task.

Other documents will also be needed to make the reference model of practical use. In particular, descriptions of how the reference model may be applied will be required, both in abstract terms and with actual examples of its use and application. To this end, a discussion

<sup>1</sup>It should be noted, however, that the PSESWG PSE reference model will expand, clarify, and update the work described in this report as they see fit for their needs.

of the possible ways of using a reference model based on these concepts is available in a companion paper [1]. Reports on the application of the reference model to actual systems will be provided in the near future.

The paper is organized as follows: the remainder of this introductory section examines the motivation for this work and outlines our view of what we mean by a PSE. Section 2 describes related work in this area, Section 3 discusses our approach to defining a PSE reference model, and Section 4 briefly explains how the model can be used. Section 5 is the main part of the paper, describing the reference model concepts in detail. Section 6 expands on issues regarding service interfaces within the reference model, and their importance with regard to representing process issues. We conclude the paper with a summary in Section 7.

## **1.1. Motivation**

A PSE services reference model is intended to help understand integration issues in a PSE and to provide a basis for identifying interface areas in which standardization may improve the ability to:

- understand, compare, and contrast available technology;
- relate users' PSE requirements to actual implementations;
- assess the ease (or otherwise) with which different PSE tools and products can be integrated.

This work should provide a better understanding of PSEs and the functionality they provide, facilitating an open environment architecture. This will create a better basis for controlling the purchase, integration, and use of development tools and products from multiple vendors.

## **1.2. Our View of a PSE**

In carrying out this work, we adopt a particular view of a PSE that affects our definition and use of the reference model. From an abstract viewpoint, we see a PSE as consisting of a *framework* populated with a set of *tools*.

The framework capabilities, which distinguish a PSE from a simple amalgamation of tools, may provide facilities to aid tool communication, provide some measure of consistency across tools, or implement commonly used operations in a single, well-defined way. Understanding, defining, and implementing framework technology has been the main goal of much of the past and current research work in the area of Integrated Project Support Environments (IPSEs), as seen in initiatives such as the Common APSE Interface Set (CAIS) [17] and the Portable Common Tool Environment (PCTE) [6].

The tools within a PSE may be many and varied, but we characterize them as being engineering tools, including both software and hardware development tools, that support some aspect

of the engineering process. We also recognize that the tools may not only be supporting the technical aspects of development, since administrative and management tools are also typical. The work on Computer-Aided Software Engineering (CASE) tools and systems has produced numerous such tools that address various aspects of systems development.<sup>2</sup>

---

<sup>2</sup>For our purposes we do not further refine the details of a PSE here. More information is available in [11].



## **2. Related Work**

We can draw from and build on a great deal of recent work when constructing a PSE reference model. We can divide this work into two broad categories:

- Existing PSEs, standards, and reference models;
- Work toward an understanding of integration.

### **2.1. Existing PSEs, Standards, and Reference Models**

We have examined a wide range of related PSEs, standards efforts, and reference models in the initial stages of our work. These include:

- The ECMA reference model for the framework of CASE environments [5], recently enhanced with additions from the National Institute of Standards (NIST) Integrated Software Engineering Environments (ISEE) working group to provide a joint NIST/ECMA reference model [10];
- The Strategic Defense System Software Engineering Support Environment (SESE);
- Engineering Information System (EIS) [7];
- Rome Airforce Base's Software Life-Cycle Support Environment (SLCSE) [15];
- DEC's Common Interface Standard (CIS) [4];
- Software Technology for Adaptable Reliable Systems (STARS) [14];
- The Common Framework Initiative (CFI);
- The Object Data Model (ODM) from the Object Management Group;
- The UNIX standardization effort, POSIX [13];
- The IEEE computer tool interconnections reference model, P1175 [8];
- TRW's Conceptual Environment Architecture Reference Model (CEARM) [12].

While each of these PSEs, standards, or reference models has interesting aspects with regard to our goals, none of them individually has the scope that we require or a definition of the concepts at a suitably abstract level. The NIST/ECMA CASE environment frameworks reference model is typical in this regard. That model characterizes framework functionality as belonging to one of six main groups—object management, process (or task) management, communication, user interface, policy enforcement, and framework administration and configuration. A number of dimensions are defined which can be used for further analyzing

environment functionality in each of those areas. The model is used as a template for examining existing environments by providing a structure, vocabulary, and categorization for discussing an environment's implementation of its functionality.

Unfortunately, while the model provides a useful basis for analyzing and comparing PSE framework mechanisms, it is not intended in its current form to help those interested in the complete PSE context such as the services available to end-users rather than just the PSE mechanisms. In our model we build upon the NIST/ECMA by using an analogous approach to consider the functionality of a populated PSE.

## 2.2. Toward an Understanding of Integration

Integration of tools in a PSE has received much attention. The traditional view of integration distinguished three independent dimensions, data, control and presentation, through which integration issues can be examined [19]. While this approach has proved useful in providing a basic understanding of the concepts, subsequent work has questioned whether these three dimensions are adequate (for example, Brown and McDermid suggest five dimensions [2]), and if the dimensions can be truly considered independently (for example, there is often a close and complex relationship between data and control integration).

As discussed by Feiler and Wallnau, the traditional view of tool integration as data, control, and presentation integration can be enhanced [18]. In addition to the original characterization, it is also useful to distinguish integration of tools with a framework and with a process, with process integration being subdivided into life-cycle process (how well the tool supports the high-level software production process being used by an organization) and development process (how well the tool supports a particular engineering activity within a life-cycle step). Framework and process integration can be seen as orthogonal to data, control, and presentation integration, the latter categories being concerned with aspects of tool to tool integration, the former concerned with the integration of a tool with its environment.

Furthermore, Thomas and Nejme [16] view integration not as a property of a component, but rather as a property of a *relationship* between components, with different properties for each relationship. They then focus on the properties of particular relationships, namely the relationship between a tool and a framework, a tool and a development process, and between one tool and another. This important concept helps to highlight the separation of integration issues not as being properties of components (and, as such, being conceived differently in each component) but as being distinct PSE characteristics in their own right.

In summary, we note that underlying current integration work is the notion that integration is an important property of a PSE, worthy of consideration in its own right. Integration is realized through framework services (defining *how* tools are implemented in terms of underlying mechanisms), in support of a development process undertaken by end-users of a PSE (defining *what* are the required semantics of the integration mechanisms being used). Separation of the semantic and mechanistic aspects of integration has been the key to a deeper understanding of the issues.

### 3. Approach Taken

Our approach to developing a PSE services reference model is a result of the study of previous work in defining reference models and standards for various aspects of PSEs, and in PSE integration. The key characteristic of previous work that has influenced our approach is the range of abstract levels at which previous models have been defined, from low level mechanistic descriptions to high level abstract architectural descriptions. This, together with our desire to build on their experiences and, where possible, to incorporate existing and proposed standards into the overall framework of our model, has resulted in the approach we have adopted.

We define a reference model based on *services*. We consider a PSE to be a collection of services, where a service can be considered to be an abstract notion that represents some aspect of PSE functionality. We make one crucial refinement of PSE services by distinguishing between *end-user services* that represent PSE functionality in terms of the services of the PSE as perceived by end-users, and *infrastructure services* that represent PSE mechanisms used to support those end-user services. Typically, for example, tool writers construct their tools using the infrastructure mechanisms of a PSE, while the tools themselves offer higher level abstract services to the PSE end-users. By taking this approach, a services-based reference model has the advantages that:

- it allows us, by focusing on the end-user view of a PSE, to present a view of PSEs that is not contingent on how the services are currently provided in the PSE. This provides separation of the concept of a set of services from a collection of mechanisms that implement those services. Hence, we can now consider different implementations for the same service, alternative implementations, adding new implementations, and so on.
- it abstracts from the notion that a PSE provides some functionality as part of the framework and other functionality within particular tools that populate the PSE. In many cases this is an artificial division based on a particular choice of PSE implementation architecture. In other cases it is a pragmatic split based on currently available software and hardware. A services-based model ensures that we are not tied to any particular current PSE architecture.
- it does not focus on a particular commercial tool, product, or environment. We do not want to presuppose an available implementation.

However, in addition to reinforcing the need for a distinction between mechanistic and semantic issues, our analysis of current work on the subject of integration in a PSE also highlighted the need for a reference model to expose the relationships between the services of a PSE as of equal importance to the services themselves. Hence, we also introduce the concept of *interfaces* and *interface areas* (i.e., collections of interfaces) as being of great importance to the reference model. There are many interfaces that must be considered in the reference model, not just the interface between two PSE services. For example, the interface between a service and an end-user making use of that service, interfaces internal to a service, and an interface between an end-user service and the framework services that support it must

## 4. Ways of Using This Reference Model

We introduce one further aspect of the model to explain the ways in which the reference model can be used. A *profile* is a characterization of an actual or proposed product (i.e., tool, framework, etc.) in terms of the elements (i.e., services and interfaces between those services) of the reference model. In some ways we can see a profile as a cross-section, or instantiation, of the PSE reference model. We can make use of the concept of a profile in a number of ways. For example:

- *To compare products.* Comparing different PSE products is difficult without a consistent conceptual framework within which to analyze all products. The description of products through profiles provides a common vocabulary for discussing them, and helps to ensure that any comparison compares "like-with-like". In addition, the categorization of services into end-user and infrastructure services means that products can be compared at different abstract levels such as their abstract functionality (end-user services) and their implementation mechanisms (infrastructure services).
- *To describe a proposed/required system.* The PSE services reference model can be used as the basis for describing a set of PSE requirements by defining a profile corresponding to a required system, as opposed to an actual system. The advantage of doing this is that the requirements can be described in an abstract way, in terms of required services and the interfaces necessary between and within those services. This is independent of particular implementation constraints, which can then be examined in the light of the abstract requirements.
- *To discuss implementation of services.* The separation of end-user and infrastructure services means that in using this model particular tools and frameworks will be characterized as providing more or less equivalent end-user services using different infrastructure services. For example, the end-user service of inter-tool communication can be realized via different infrastructure services—a remote procedure call mechanism, message server facilities, data sharing with triggers, and so on. For each possible implementation a different profile can be defined. Hence, different ways of implementing common end-user functionality can more easily be represented and analyzed.
- *To examine product integration issues.* Profiles of actual PSE products characterize both their abstract functionality and implementation mechanisms. When users wish to determine the extent to which those products can be integrated, the profiles provide the necessary basis for answering important questions regarding the ease with which the integration can take place. For example, the end-user service aspects of the profiles can reveal the extent to which the products provide similar services, while the infrastructure aspects allow issues of mechanism interoperation to be discussed. Hence, a PSE integrator may use the PSE services reference model to determine, for a collection of tool products to be integrated, what services each



tool provides and, based on the overlap of provided services and available base computing environment, to develop a strategy for integration in terms of a particular environment architecture, identifying interfaces relevant for its realization.

- *As an aid to identifying PSE interface areas where standards are available.* It would be of great benefit to many PSE tool writers, environment builders, and end-users if standards were available within the PSE area. The problem is in identifying the interfaces in a PSE where standardization would help, selecting (or developing) standards at those interfaces, understanding the relationship between those interfaces (and therefore between the standards defined at those interfaces), and evaluating how particular tools and products adhere to those standards. Having identified services in the reference model, the aim is to then see how those services relate, and which interfaces between the services are relevant. Profiles based on a selection of standards at those interfaces can then be discussed and evaluated.

In summary, we note that in presenting the main elements of a PSE services reference model we abstracted from a notion of tools and frameworks towards the higher level concept of services and interfaces. Now, in examining an actual tool or PSE product, the reference model is used to reflect issues of functionality and architecture by allowing an abstract description of that product to be produced. The concept of a profile is the basis for this task, characterizing a product in the context of the elements of the reference model. Profiles can provide the basis for analysis, comparison, and evaluation of (actual or proposed) PSE systems and products.<sup>1</sup>

---

<sup>1</sup>Profiling issues are dealt with in much greater detail in a companion paper which provides detailed descriptions and examples of the use of profiling as an analytical technique [1].

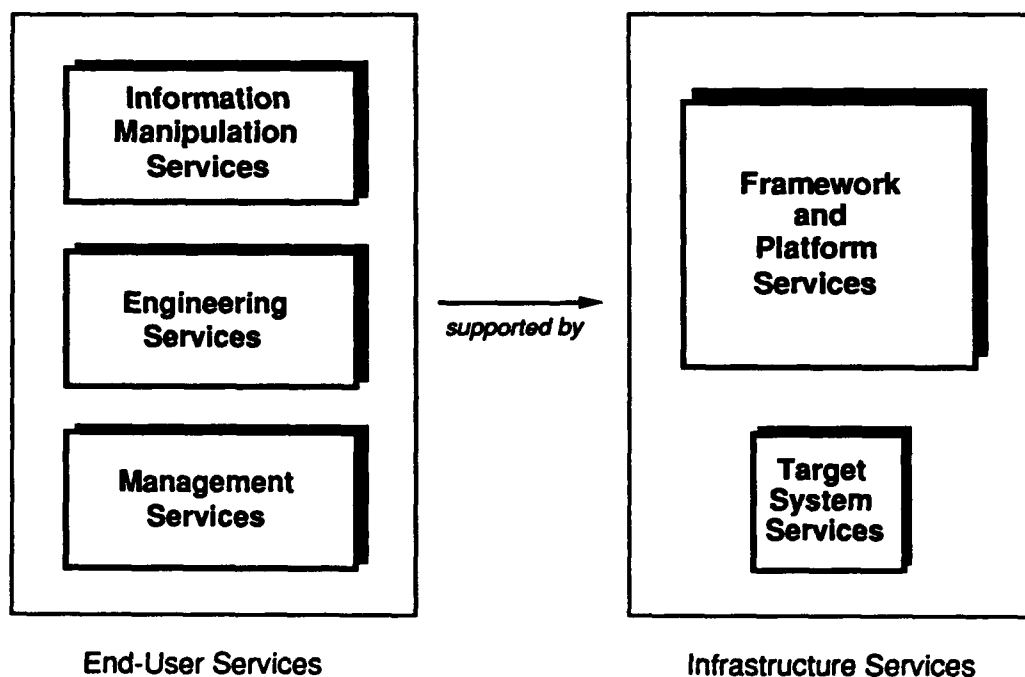
## 5. The Basic Concepts of the Model

The reference model is a services-based reference model. By this we mean that the view of a PSE presented considers a PSE as a collection of services, partitioned into end-user services and infrastructure services. An end-user service may be supported by one or more infrastructure services. As a result, the reference model can be used for both comparison and analysis of available services in a PSE, both end-user services and infrastructure services, and to relate end-user services to different implementations of those services.

In this section we describe a selection and grouping of services into service areas that we believe are useful as the basis for a PSE services reference model. For a complete reference model description each of these service areas should be described in detail, identifying the particular services within each area. In Appendix A, we provide a more detailed list of services, collated from the reference models and standards we surveyed in the initial stages of our work.

### 5.1. The PSE Services Reference Model

A graphical depiction of the main elements of the PSE services reference model, shown in



**Figure 5-1 The Main Components of the PSE Services Reference Model**

Figure 5-1, is arranged such that services providing the mechanisms, or infrastructure, for a PSE implementation are shown on the right hand side, and end-user services that represent

PSE functionality as seen by the end-users are shown on the left. The distinction being made is that the infrastructure services support the services perceived by the PSE end-users.

### 5.1.1. PSE Infrastructure Services

The PSE infrastructure is characterized as consisting of target system services and a set of framework and platform services, as illustrated in Figure 5-2.

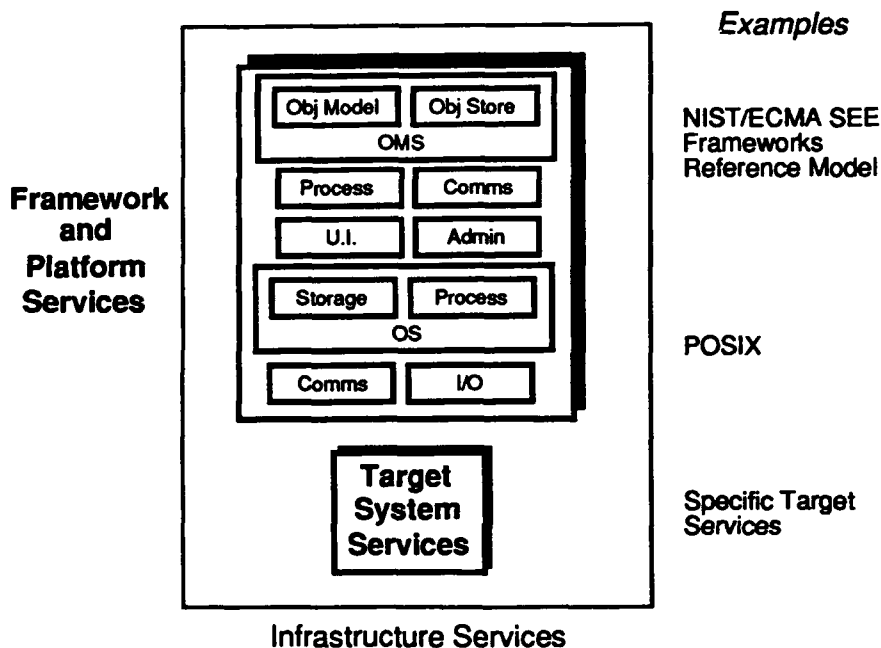


Figure 5-2 The PSE Infrastructure Services

The *target system service* is the computing system on which the application being developed under a PSE will eventually operate. Distinguishing target services helps represent the situation in which different facilities are available for the manipulation of application programs on the host and target environments. These may differ from the services available on the host environment. The target system services may consist of heterogeneous components (e.g., the target system services may consist of different vendor hardware and different operating system software).

The main set of infrastructure services can be described as *framework and platform services*. The purpose of the platform services is to provide a virtual machine layer that hides discrepancies between different host or target system components. As a result, services interfacing to the host or target system services through the platform services are isolated from those differences, making them more portable across different instances.

While the platform services provide general computing facilities for every computer user, environment framework services are considered to be additional facilities more targeted towards

PSE support. We divide the environment framework services into several service areas. Following the NIST/ECMA CASE environment frameworks reference model, the service areas we define are data storage and data modeling (two aspects of what is called "object management" in NIST/ECMA), process management, user interface, and communication.<sup>1</sup>

It is important to note that the distinction between framework and platform services is not always obvious. In particular, services that are currently thought of as at the framework level (e.g., aspects of the object management services) may in the future be provided as part of the platform. In addition, each of the service areas may provide services at several levels of abstraction within that area, or alternative services within a single area. We illustrate this with two examples:

1. At one abstract level within the process management area, support may be provided for a non-persistent process concept as typically found in an operating system, while at another level, support for persistent processes that may be enacted by humans (i.e., tasks) may be supported through event triggering and notification mechanisms.
2. The data modeling service area may offer relational, Entity-Relationship (ER), and Object-Oriented (OO) meta models, each of which is constructed from the file-based storage mechanisms provided by the operating system.

For some aspects of the environment framework services such as versioning and security services, it is not immediately clear in which of the five areas they belong. As in the NIST/ECMA model, we treat low-level versioning services and aggregation/configuration services as part of the data storage service area. Security is a more difficult concept. In some architectures it is an integral part of data storage (e.g., in capability based systems everything, including processes, is a data object, accessible only through capability objects), while in other architectures security is seen as an "add-on" provided in different services (e.g., as part of process support as well as part of a file system directory service). In the context of a service reference model, our first inclination is to treat security primitives as a service that is part of an administrative service area.

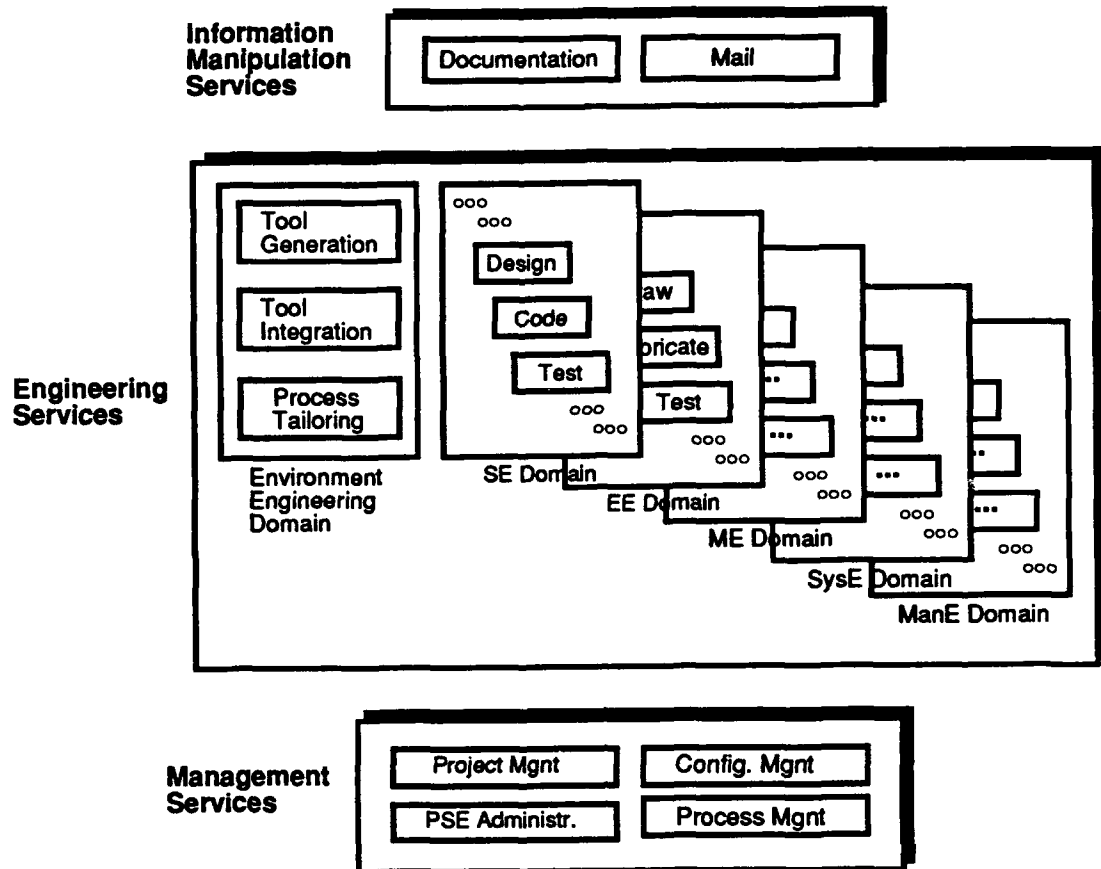
### 5.1.2. PSE End-User Services

PSE end-user services are those facilities provided directly to end-users of a PSE. These services can be divided into three categories—information manipulation services, engineering services, and management services—as illustrated in Figure 5-3.

In many aspects of systems development *information manipulation services* are required to support document generation, update and display, and for informal communication between PSE users. These are general support services that are independent of a particular application domain for the PSE.

---

<sup>1</sup>While we follow the divisions defined in the NIST/ECMA reference model, we introduce new terminology for the object management area to draw a distinction between the storage and modeling services provided to emphasize that different modeling approaches can be supported by the same storage services, and different storage services can support the same set of modeling services.



**Figure 5-3 The PSE End-User Services**

The *engineering services* provide support for the technical aspects of engineering activities. These can be further subdivided into services for specific engineering domains such as software engineering (SE), electrical engineering (EE), mechanical engineering (ME), systems engineering (SysE), and manufacturing engineering (ManE). From a simplistic view, within each such engineering domain the engineering life-cycle defines a number of engineering activities, each of which requires services for its support. For software engineering, the activities are often defined as requirement analysis, design, code, test, documentation, and so on. Engineering activities primarily reflect development processes, while each engineering domain and collection of engineering services support the life-cycle process for that domain. Each engineering activity is represented in the model by a service group. Figure 5-3 illustrates a number of possible groups. In addition, more general services may be required that span life-cycle phases, or cross a number of engineering domains. For example, many aspects of the services for gathering and analyzing of requirements may be common to all of the engineering domains.

The engineering services discussed so far are aimed at users of populated PSEs (i.e., a framework populated with specific tools). It is useful to also recognize another type of PSE user, the PSE builder and maintainer, that supports the building, integrating, adapting, and

monitoring of a PSE. The environment engineering services support the PSE builders and maintainers. These services subdivide into tool generation services (i.e., tool construction using tool generation systems), tool integration services (i.e., services for adapting tools for use in a PSE, and for upkeep of a populated PSE), and process tailoring services (i.e., enabling end-user services to be tailored to different process needs by providing access to tools for different classes of PSE end-users, and so on).

*Management services* provide support for the managerial aspects of the engineering activities. These services subdivide into the project management, PSE administration, configuration management, and process management areas. Within the project management domain we can distinguish project planning and control, project configuration management, project process management, and so on. The way in which the management services are used for a particular PSE instance will be determined by the particular project organization and project management model being enacted.

Within these services are a number of facilities to aid project development. These include:

- Services for each of the steps in the project planning and control sub-domain. These are typically services that support an individual project manager (e.g., WBS, scheduling, resource assignment, etc.).
- Services for project configuration management, including problem report handling, change management, repository and developer workspace management, and developer team support.
- Services for project process management, including facilities for process description, enactment, and measurement.

A further important aspect is the PSE administration services. These services are the reflection of project management information in the populated PSE (e.g., access control based on task assignments), and provide support for the administration of computing resources (e.g., tool licensing and installation, utilization and reassignment of processor and memory resources, etc.).



## 6. Toward an Understanding of Service Interfaces

Having described the services provided in a PSE, we now focus on the interfaces between those services. We refer to an *interface area* as the collection of interfaces possible between a set of services. To realize interaction between services we must distinguish both the semantic aspects of the interface (i.e., the process aspects), and the mechanistic aspects (i.e., the use of different framework services to provide those semantics).

### 6.1. PSE End-User Service Interfaces

Interfaces between end-user services reflect the semantics of interaction between those services. Typically, these interfaces can be expressed in a number of forms, such as domain data models, service functions requestable by end-users or by other services, or behavioral models describing the behavior of the service based on external input. These interfaces may be provided by the infrastructure services in different ways, depending on the architecture of the PSE. For example, data in a domain data model may be accessible through a common data storage service, or may be imported and exported from one end-user service to another via a communication service.

At the external level of a populated PSE, it is ultimately the PSE end-user who interacts with the end-user services. It is the user interface services that are mainly responsible for implementing this interface to the end-users.

An important point to emphasize is that within the end-user services there are a number of different interface areas that may be of interest. We describe elements of this range by first considering interfaces within one service, then between services. The list focuses on the semantic aspects of the interface. Of interest are:

- Interfaces that permit two instances of one end-user service (e.g., different ER design editors, or different C compilers) to interoperate, or to allow one instance to be replaced by another.
- Interfaces that permit different services within one life-cycle step to work together (e.g., cooperation between compilers and debuggers, or between ER design editors and design analyzers).
- Interfaces between two specific steps in a life-cycle domain. These are interfaces that typically require some form of mapping or transformation to convert representations defined in one step into those required by another. These can be interfaces within an engineering domain (e.g., a mapping between identifiers in the coding language and the design language), or interfaces within a management domain (e.g., a mapping between WBS and schedule elements).
- Interfaces to an engineering domain, or to the management domain. These would define the use of the services within the context of higher level organizational policies. Particular life-cycle and project models would be



represented as data models and behavioral models within the PSE. In fact, services from individual engineering steps, as well as management services, may interface to the life-cycle model. Similarly, individual management services, as well as engineering services, may interface to the project model.

- Interfaces between management and engineering services. Typically, management services must interact with individual engineering services, and engineering domains. For example, performing the management service of metrics analysis requires the collection of metrics information within individual engineering steps, together with metrics gathering at the life-cycle level.

In summary, we see that there are many interface areas of interest within the PSE end-user services. Each of these will be of interest to different PSE users.

## **6.2. PSE Infrastructure Service Interfaces**

Within the PSE infrastructure services there are also interface areas that are of interest to us. As seen in the end-user services, different categories of interface exist both within a single service and between different services. For example, there are interfaces between two instances of a framework service (i.e., interoperability between possibly non-identical service instances), between framework services and platform services, as well as interfacing between services within one service area and between service areas.

In examining a particular PSE, the architecture and profile of infrastructure services within that PSE will determine the specific interconnections between services, and therefore the interfaces that are relevant. The PSE reference model will help in identifying and understanding the relationships between those interfaces. For example, the platform services in a particular PSE may consist of a distributed computing environment realized as a communication service. At the framework services level, a distributed transparent object management service may be provided, implemented on top of the platform communication service.

Finally, the end-user services must interface to the infrastructure services. It is possible that end-user services for a particular PSE may interface only to framework services, to platform services, or even directly to host or target system services. For example, a particular engineering tool may not utilize a meta data model to explicitly express a domain data model, and therefore neither requires access to such services externally, nor provides an interface to such services. In this case, the tool may directly interface to the host or target system services. An understanding of particular platform, framework, and tool architectures is necessary to guide the PSE builder in deciding which interfaces are relevant.

### 6.3. Tools and Environment Architectures

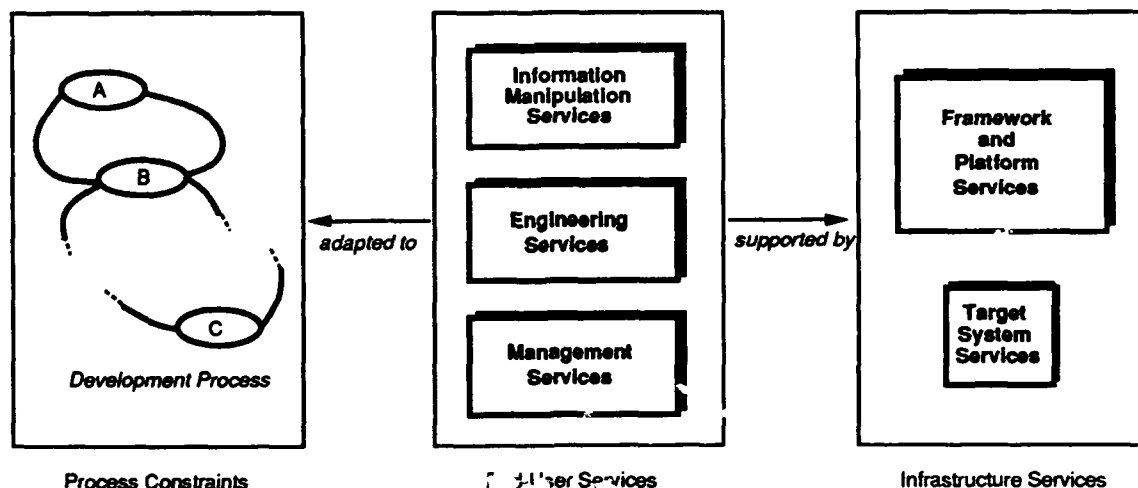
The PSE services reference model, and the interface areas identified by it, should be able to accommodate a number of different tool and environment architectures, and tool integration approaches. This is important as there are many different possible realizations of PSE services. In particular, tools and environments may cooperate in different ways to provide some, or all, of the services defined. For example, an individual tool product typically provides a number of different services within the one tool. Often it may only offer an interface to a subset of those services. Similarly, tools may implement those services in different ways. For example, a tool may implement all of its services itself and only rely on platform services to support its operation, or it may utilize other tools and services in its implementation (e.g., a relational database).

There are also a variety of approaches towards the actual interoperation of mechanisms between tools. One common way to allow tool interoperation is through the tools' use of the same instance of a data storage service (e.g., an ER data model provided as part of the platform services) and the same user interface management system (e.g., the Motif/X window system provided as part of the user interface services). However, even when sharing these mechanisms there are different possible levels of interaction. For example, even though two tools may use the same instance of the data storage service with an ER data model, they may not have an agreed domain data model. Similarly, even having agreed upon a domain data model, one tool may be using an ER conceptual schema with an underlying relational implementation, while the other maintains the data internally in an object-oriented database. Each of these alternatives offers a different degree of integration and utilizes a different set of interfaces. The integration properties as discussed by Thomas and Nejme [16] (e.g., redundancy of service between tools and framework) are one attempt to characterize these different forms of integration.

A major attraction of a services-based PSE reference model is that when considering the integration of a number of tool products with a particular framework product, the PSE service reference model can help determine the amount of overlap in providing the same services, identify the sets of interfaces provided by the products, look for patterns of commonality between the tool services, and derive an integration strategy that takes into account integration alternatives, desired degree of tightness of integration based on the process to be supported, and the need for flexibility to accommodate product heterogeneity, product replacement, and adaptability of process over time.

### 6.4. Process Issues

Knowledge of the development process being used within an organization is essential for understanding the way in which end-user services of a PSE interact. We can view the development process as constraining meaningful interaction of services. This knowledge not only guides our understanding of the end-user services of interest, it also helps us to focus on interfaces between services that are important to a particular organization.



**Figure 6-1 Adding Process Aspects to the PSE Services Reference Model**

Another way to view this relationship between end-user services and development process, as illustrated in Figure 6-1, is to consider the end-user services to be *adapted* within the constraints imposed by a particular development process.

In most existing PSEs, knowledge of the development process is implicitly buried within the tools and services provided through the constraints imposed on the ordering of tool invocations, the sharing of information between tools, and the expected usage patterns of the tools. More explicit process support remains at the level of manual procedures on the use of the PSE, as found in company standards and procedure manuals. As interest in understanding and modeling the development process has increased, so have the number of attempts at supporting and automating aspects of the process within PSEs through various process modeling languages and formalisms. The PSE reference model presented here recognizes the importance of this work, providing a context for discussing these issues.

Indeed, we can view the move towards explicit process support as part of a trend in automated development support. Initially, such support was simply at the platform level, with operating system facilities providing the only tools available to programmers. Initiatives such as the Ada programming support environment (APSE) [3], the common APSE interface set (CAIS) [17], and the portable common tool environment (PCTE) [6] helped to focus attention on the framework services required for more extensive developer support. Currently, the numerous tools available in the marketplace have led to interest in how the services provided by those tools can be combined with the framework services to provide a complete PSE. In the future, this work is likely to recognize that process concerns play a major role in understanding, adapting, and integrating the tool services for a particular development organization. Explicit support for process representation and enactment will be an essential component of future PSEs.

## 7. Summary and Conclusions

This paper has described in outline a PSE services reference model for the purpose of characterizing PSEs through distinguishing the services they make available and the interfaces provided within and between those services. The main characteristics of the model are:

- The separation of end-user services, as perceived by PSE end-users, and infrastructure services, which support those end-user services. These are constrained with the context of the development process being followed by a particular organization.
- A framework that can reflect current PSE standardization efforts, identifying the areas in which sufficient work is taking place and those in need of most attention.
- The definition of a base model with ample scope for future development to suit new requirements within particular application areas.

As a result, we believe that the proposed PSE services reference model provides a useful basis for further investigation, containing many of the elements desirable to meet our stated aims.

It should be emphasized, however, that this paper represents our current thoughts on the elements necessary in defining a PSE reference model. These thoughts are still evolving and should not be taken as a final statement. Hence, comments, suggestions, and criticisms of this work are encouraged.

## Acknowledgements

The roots of this work are in various environment related projects at the SEI and other places. In particular, interactions with members of the Software Development Environment Project and CASE Technology Project at the SEI, Tricia Oberndorf from NADC, Dave Carney from IDA, and participants of NIST ISEE and NGCR PSESWG have all contributed to the work reported here.

We are grateful for comments on previous drafts of this paper from Dave Carney, Ed Morris, Tricia Oberndorf, and Kurt Wallnau.



## **Appendix A Detailed Breakdown of PSE Services**

In this appendix we provide a detailed breakdown of PSE services. This is not intended to be a definitive list of PSE services. Rather, it is an illustration of a more comprehensive list of services that may be accommodated within the framework of the PSE services reference model described in this paper.

The services presented in this appendix have been derived from the list of existing and proposed standards and reference models in the general area of PSEs described earlier in the paper.

In the list that follows, indentation implies a lower level of detail in this services breakdown.

### **Framework Services**

#### **Data Management and Data Model**

- Persistent data storage
- Relationship management
- Naming
- Data distribution
- Data transaction
- Concurrency
- Process support
- Archiving
- Backups
- Version management
- Configuration management
- Derivation histories
- Query facility
- Metadata
- State monitoring
- Views
- Navigation
- Data replication and synchronization
- Access control and security
- Constraint enforcement
- Function attachment
- Global schema management

#### **Process/Task Management**

- Task definition
- Task execution
- Task transactions
- Task history
- Event monitoring
- Auditing and accounting
- Role management
- Constraint support

## **User Interface**

- Dialog management**
- Markup languages**
- Multi-media**

## **Communication**

- Message service and delivery**
- Tool registration**

## **Administration**

- Metriation**
- Security control**
- Security monitoring**

## **Platform Services**

### **Data Storage**

- Non-persistent memory management (physical and virtual)**
- Persistent memory management**

### **Communication**

- Lower levels of ISO OSI model**

### **I/O**

- Device access**
- Keyboard management**
- Bitmap screen handling**
- etc...**

### **Process**

- Operating system process management**

### **Data model**

- ASCII**
- ISAM**

## **Management Services**

### **Project Management**

- Project planning (WSBS, scheduling, resource allocation)
- Project control (monitoring, reporting, engineering process control)
- Configuration management (change and release management, team support)

### **Process Management**

- Process quality assurance enforcement
- Process measurement
- Process improvement

### **PSE Administration**

- Plan-based security and authorization
- Service and tool management (installation, licensing)
- Resource management (accounting, load balancing)

## **Engineering Services**

### **Software Engineering Process**

- Requirements definition
- Specification
- Design
  - creation
  - analysis
  - simulation
  - code generation
- Coding
  - creation
  - analysis
  - translation
  - debugging
  - design generation
- Testing
- Verification (analysis of specifications, designs, etc...)
- Validation (design and coding standards, etc...)
- Documentation
  - user manuals
  - reference manuals
  - reference model description (!)

### **Electrical Engineering Process**

.....  
.....



## **Mechanical Engineering Process**

.....  
.....

## **PSE Maintenance Services**

### **Tool Generation**

**Parsers**  
**Transformers**

### **Tool Integration**

**Encapsulation (tool wrappers)**  
**Adaption (parameterization, configurations)**  
**Version management**

### **Process Tailoring**

# Glossary

reference model	An abstract description of a system that can act as a reference point for discussion, analysis, and comparison.
service	A collection of PSE functionality.
infrastructure	The software and hardware components of a PSE that provide the basic mechanisms for communication and coordination between tools.
end-user service	A service available for direct interaction by end-users of a PSE.
infrastructure service	A service that is provided to support, or implement, some aspect of one or more end-user services
profile	A characterization of a particular PSE tool or system in terms of the elements of the PSE services reference model.
PSE services reference model	An abstract description of a PSE based on describing a PSE as a collection of services. The description is capable of being used for discussion, analysis, and comparison of current and proposed PSE tools and products.



## References

- [1] A.W. Brown and P.H. Feiler. *Using a Project Support Environment Services Reference Model*. Technical Report CMU/SEI-92-TR-3, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, January 1992.
- [2] A.W. Brown and J.A. McDermid. On Integration and Reuse in a Software Development Environment. In F. Long and M. Tedd, editors, *Software Engineering Environments '91*, Ellis Horwood, Chichester, England, 1991.
- [3] J.N. Buxton. *Requirements for APSE – Stoneman*. U.S. Department of Defense, February 1980.
- [4] Digital Equipment Corporation. *CASE Interface Service Base Document, V1.0*, September 1990.
- [5] ECMA. *A Reference Model for Frameworks of Computer-Assisted Software Engineering Environments*. Technical Report TR/55, European Computer Manufacturers Association, Geneva, Switzerland, December 1990.
- [6] ECMA. *Portable Common Tool Environment (PCTE)*. Technical Report ECMA-149, European Computer Manufacturers Association, Geneva, Switzerland, December 1990.
- [7] Honeywell. *EIS Specification: Volumes 1, 2, and 3*. Honeywell, Minneapolis, MN, 1990.
- [8] IEEE. A Standard Reference Model for Computing System Tool Interconnections. Technical Report Draft P1175/D11, IEEE, New York, NY, May 1991.
- [9] NGCR PSESWG. A Reference Model for Project Support Environment Standards. Technical Report Draft, US Navy NGCR Programme, January 1992.
- [10] NIST/ECMA. A Reference Model for Frameworks of Software Engineering Environments (Version 2). *ECMA Report Number TR/55 (Version 2)*, NIST Report Number SP 500-201, December 1991.
- [11] P.A. Oberndorf. *The Scope of an NGCR Project Support Environment*. Technical Report unnumbered, NGCR PSESWG, November 1991.
- [12] M.H. Penedo. *A Survey of Software Engineering Environment Architectural Approaches*. Technical Report Arcadia-TRW-90-004, TRW, Redondo Beach, CA, December 1990.
- [13] POSIX. Standards Project: Draft Guide to the POSIX Open Systems Environment. Technical Report P1003.0/D14, IEEE, Institute of Electrical and Electronics Engineers, New York, November 1991.
- [14] STARS. *Proceedings of STARS'91*. STARS, Annapolis, MD, December 1991.

- [15] T. Strellich. The Software Life Cycle Support Environment (SLCSE): A Computer Based Framework for Developing Software Systems. In *Proceedings of the ACM SIGSOFT/SIGPLAN Software Engineering Symposium on Practical Software Development Environments*, Boston, MA, November 1988.
- [16] I. Thomas and B.A. Nejmah. Definitions of Tool Integration for Environments. *IEEE Software*, 9(2):29–35, March 1992.
- [17] U.S. Department of Defense. Common APSE Interface Set (CAIS) – Revision A, May 1988.
- [18] K.C. Wallnau and P.H. Feiler. *Tool Integration and Environment Architectures*. Technical Report CMU/SEI-91-TR-11, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, May 1991.
- [19] A. Wasserman. Tool Integration in Software Engineering Environments. In F. Long, editor, *Software Engineering Environments*, number 467 in Lecture Notes in Computer Science, pages 138–150, Springer-Verlag, Berlin, Germany, 1990.

## REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION <b>Unclassified</b>			1b. RESTRICTIVE MARKINGS <b>None</b>		
2a. SECURITY CLASSIFICATION AUTHORITY <b>N/A</b>			3. DISTRIBUTION/AVAILABILITY OF REPORT <b>Approved for Public Release Distribution Unlimited</b>		
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE <b>N/A</b>					
4. PERFORMING ORGANIZATION REPORT NUMBER(S) <b>CMU/SEI-92-TR-2</b>			5. MONITORING ORGANIZATION REPORT NUMBER(S) <b>ESD-92-TR-2</b>		
6a. NAME OF PERFORMING ORGANIZATION <b>Software Engineering Institute</b>		6b. OFFICE SYMBOL (if applicable) <b>SEI</b>	7a. NAME OF MONITORING ORGANIZATION <b>SEI Joint Program Office</b>		
6c. ADDRESS (City, State and ZIP Code) <b>Carnegie Mellon University Pittsburgh PA 15213</b>			7b. ADDRESS (City, State and ZIP Code) <b>ESD/AVS Hanscom Air Force Base, MA 01731</b>		
8a. NAME OFFUNDING/SPONSORING ORGANIZATION <b>SEI Joint Program Office</b>		8b. OFFICE SYMBOL (if applicable) <b>ESD/AVS</b>	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER <b>F1962890C0003</b>		
8c. ADDRESS (City, State and ZIP Code) <b>Carnegie Mellon University Pittsburgh PA 15213</b>			10. SOURCE OF FUNDING NOS.		
			PROGRAM ELEMENT NO <b>63756E</b>	PROJECT NO. <b>N/A</b>	TASK NO <b>N/A</b>
			WORK UNIT NO. <b>N/A</b>		
11. TITLE (Include Security Classification) <b>The Conceptual Basis for a Project Support Environment Services Reference Model</b>					
12. PERSONAL AUTHOR(S) <b>Alan W. Brown and Peter H. Feiler</b>					
13a. TYPE OF REPORT <b>Final</b>		13b. TIME COVERED FROM TO		14. DATE OF REPORT (Yr., Mo., Day) <b>January 1992</b>	
15. PAGE COUNT <b>34 pp.</b>					
16. SUPPLEMENTARY NOTATION					
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)		
FIELD	GROUP	SUB. GR.	integration project support environment services reference model		
19. ABSTRACT (Continue on reverse if necessary and identify by block number)					
<p>The foundation for a Project Support Environment (PSE) services reference model is presented. This model is to be used as the basis for understanding more about the meaning of integration in a PSE, comparing and contrasting PSE tools and products, and for helping in the identification of PSE interface areas that are candidates for standardization.</p> <p>The model views a PSE as a set of services, distinguishing between services as perceived by PSE end-users, and those provided as mechanisms within the PSE infrastructure. Process constraints on those services are separately identified. The motivation for the view of a PSE is described, followed by a detailed description of the main structure and elements of the model.</p> <p style="text-align: right;">(please turn over)</p>					
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT <b>UNCLASSIFIED/UNLIMITED SAME AS RPTDITC USERS</b>			21. ABSTRACT SECURITY CLASSIFICATION <b>Unclassified, Unlimited Distribution</b>		
22a. NAME OF RESPONSIBLE INDIVIDUAL <b>John S. Herman, Capt, USAF</b>			22b. TELEPHONE NUMBER (Include Area Code) <b>(412) 268-7631</b>		22c. OFFICE SYMBOL <b>ESD/AVS (SEI)</b>

STRACT —continued from page one, block 19